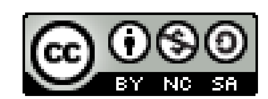
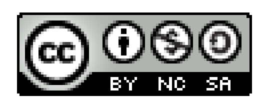
Servidor funcional basado en Raspberry PI con panel de gestión centralizada.

Antonio Mónaco Osado

Indice

Panel de gestión de servicios basado en Raspberry PI




BY NC SA

1 Justificación

Este proyecto está hecho pensando en las microempresas, a las que normalmente no se les ofrecen servicios pensados para ellas específicamente.

Se suele dar servicio a pequeñas, medianas y grandes empresas. Las microempresas deben escoger entre dispositivos para el hogar, que normalmente carecen de las funcionalidades necesarias, o para pequeña/mediana empresa, y estos últimos suelen ser bastante caros.

He querido ofrecer una solución a ese problema implementando este proyecto, además basado en Hardware Abierto y Software Libre, evitando que los clientes paguen licencias, reduciendo el coste final y permitiendo que cualquier persona que lo desee mejore el sistema.

Además el reducido tamaño de todo el sistema, y su bajo consumo eléctrico permiten su alojamiento en cualquier negocio por pequeño que sea el emplazamiento del mismo, sin grandes facturas eléctricas.

Este proyecto abarca todas las asignaturas que hemos visto estos dos años en el CFGM de SMIX.

Primero lo he instalado y configurado en una o varias máquinas virtuales, haciendo todo el trabajo con scripts, para finalmente implementarlo en la Raspberry.

Y a eso se debe que esté así distribuido.

2 Que pretendo hacer?

La idea principal es la de diseñar el complemento que convierte un router convencional de cualquier ISP en una red con todo, o casi todo lo necesario para informatizar una pequeña empresa.

Este accesorio debe asignar direcciones IP por DHCP a los clientes, asignándoles un servidor DNS local que también se alojara en nuestra raspberry, por lo tanto deberemos configurar el router para que redireccione las peticiones DHCP si este lo permite o deberemos dividir las IP's de la red en dos rangos.

Uno para la pi, y otro para el router, y asignar algunas IP's estáticas para nodos fijos como ahora el router o la pi. He escogido ISC-DHCP-SERVER y Bind9, ya que los hemos aprendido a usar en clase y me siento familiarizado con estos dos servicios, además hara de servidor de correo y dispondrá de una interfaz webmail gracias a postfix, courier y roundcube, vistos también este año en clase.

La raspberry también actuará como servidor de impresión y escaneo, disponiendo de la opción de escanear directamente en nuestro PC en PDF, JPG, etc desde un software cliente, para permitir el escaneo desde cualquier sistema operativo, e imprimir a PDF, pudiendo recoger los documentos impresos en un directorio compartido por un protocolo multiplataforma lo más fácil de implementar posible por el lado del usuario, en este caso SAMBA.

Intentando evitar en la medida de lo posible deber instalar aplicaciones cliente, y aprovechando los protocolos y servicios que vienen preinstalados en los sistemas operativos de los clientes a los que se les ofrecerá este producto.

Por eso he escogido SAMBA para compartir directorios y los drivers de las impresoras CUPS para que los clientes Windows puedan utilizarlas.

Como servidor de impresión he escogido CUPS, ya que es compatible con Windows, Mac y Linux y dispone del módulo cups-pdf para probar el sistema con una impresora virtual. Para hacer de servidor de escaneo he escogido SANE, y para el correo Postfix y courier, finalmente para el webmail Roundcube.

Será necesario configurar un cliente en cada plataforma escogida para probar la funcionalidad de este modulo acoplable a cualquier router.

Como router he escogido un router Linksys que tenia en casa con firmware Linux gracias a la versatilidad y funcionalidades que ofrece, y la total integración que permite con este proyecto.

Los clientes deben disponer de una interfaz totalmente intuitiva para administrar tanto la máquina servidor como sus servicios y aplicaciones desde cualquier máquina de la red, para poder guiar por teléfono a los clientes al realizar cualquier tipo de reparación rutinaria, por lo que he escogido Webmin con una interfaz amigable, y he creado un usuario que solo tiene acceso a las opciones necesarias para configurar lo que nosotros vamos a ofrecer.

Webmin debe ser de acceso desde el exterior para poder actuar cuando la incidencia se complica, o si el usuario en cuestión no llega a entender el proceso a realizar, o fuera demasiado complejo para alguien sin los conocimientos necesarios.

Tambien debemos disponer de una conexión segura hacia una terminal en el servidor, así que implementaré SSH con autentificación con clave pública-privada, entre otros métodos para securizar el servicio y automatizar los backups.

Además como el acceso es desde el exterior, y no dispongo de un dominio accesible desde internet, iva a utilizar no-ip para probar que lo que he implementado funciona, ya que DynDNS es de pago desde hace un tiempo y el método es exactamente el mismo.

Como desde el instituto esto no es posible, ofrezco la alternativa de una simulación con una máquina virtual que hará de “Hosting en Internet”, y a su vez de técnico helpdesk conectandose a la Pi.

Finalmente, para control remoto gráfico a nivel local, he decidido instalar un servidor Free RDP, ya que también es multiplataforma y además gráfico, para que cualquier usuario de una empresa cliente pueda gestionar la raspberry desde su estación de trabajo de una forma totalmente gráfica.

Si da tiempo implementaré la funcionalidad de RDPS, tunelizando esa conexión con SSH para las conexiones desde fuera de nuestra red.

Adicionalmente, he instalado ckermit para que el sistema pueda usarse como consola para routers Cysco o HP ProCurve. Puede controlar cualquier dispositivo por puerto Serie.

He decidido utilizar la mayor parte de software libre o al menos abierto, además de un hardware abierto como Raspberry PI, por la filosofía que esto implica.

Para la pagina de inicio que será el panel central he decidido crear una página HTML y su archivo css, para lo que primero hice un “croquis” en KolourPaint para decidir de una forma mas visual que es lo que quería exactamente.

Panel de gestion de servicios





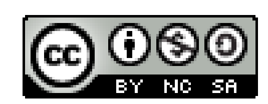








Antonio Mónaco Osado



Finalmente hará falta implementar copias de seguridad periódicas, y no hará falta ningún documento, ya que no alojamos datos de carácter personal. Si la empresa cliente los aloja, es esa su responsabilidad. Estos backups los haré utilizando las opciónes que Webmin nos otorga, pero para según que información sensible utilizaremos cron y anacron, para cifrar los backups con GNUPG.

3 Que es Raspberry pi?

Raspberry Pi es una pequeña placa computadora de bajo coste desarrollada en Reino Unido por la Fundación Raspberry Pi con el objetivo de estimular la enseñanza de ciencias de la computación en las escuelas, es tan pequeño como una tarjeta de crédito, y funciona con un mínimo consumo eléctrico.

Este pequeño ordenador es excelente para desarrollar software, es una herramienta ideal para aprender programación, y administración de sistemas informáticos y redes.

4 Características técnicas de Raspberry PI

El tamaño de Raspberry Pi es de 8,5 x 5,3 cm, por lo tanto no supera el perímetro de una tarjeta de crédito.

Si abrimos su carcasa, encontramos un chip integrado broadcom BCM2835 SoC, que contiene un procesador ARM11 a 700 MHz, con posibilidad de overclock hasta 1 GHz y un chip de 512 MB dememória RAM.

Junto al chip SoC, encontramos un procesador gráfico (GPU) videocore IV.

La tarjeta SD tiene que ser mínimo de 4 GB ya que las distribuciones se entregan en archivos con extensión .img de 4GB, y a ser posible debe ser una SD HC, ya que el sistema operativo debe correr con la mayor fluidez posible.

La alimentación se realiza mediante una conexión micro USB, con el mismo cabezal que utilizan la mayor parte de los cargadores para Smartphones actuales (debe tener un voltaje de 5V a 1,5A), y también se puede alimentar a través de un periférico alimentado conectado a el por USB.

En la placa nos encontramos además con una salida de vídeo y audio a través de un conector HDMI, lo que nos permite conectar la pi tanto a televisores, como a monitores que cuenten con dicha conexión. En cuanto a vídeo se refiere, también cuenta con una salida de vídeo compuesto y una salida de audio minijack de 3,5mm.

5 Por que Raspberry?

He utilizado Raspberry Pi, para dar a conocer el Hardware Libre, y demostrar que con el, y con tiempo y dedicación se pueden hacer grandes cosas por uno mismo, aprovechando lo que otra persona ha pensado, para hacer algo más grande, y/o adaptado a mis necesidades, pero siendo funcional y estable.

Siguiendo el refrán que dice para qué reinventar la rueda si solo quiero hacer un coche.

6 Diferencias entre sus dos modelos

6.1 El modelo "A"

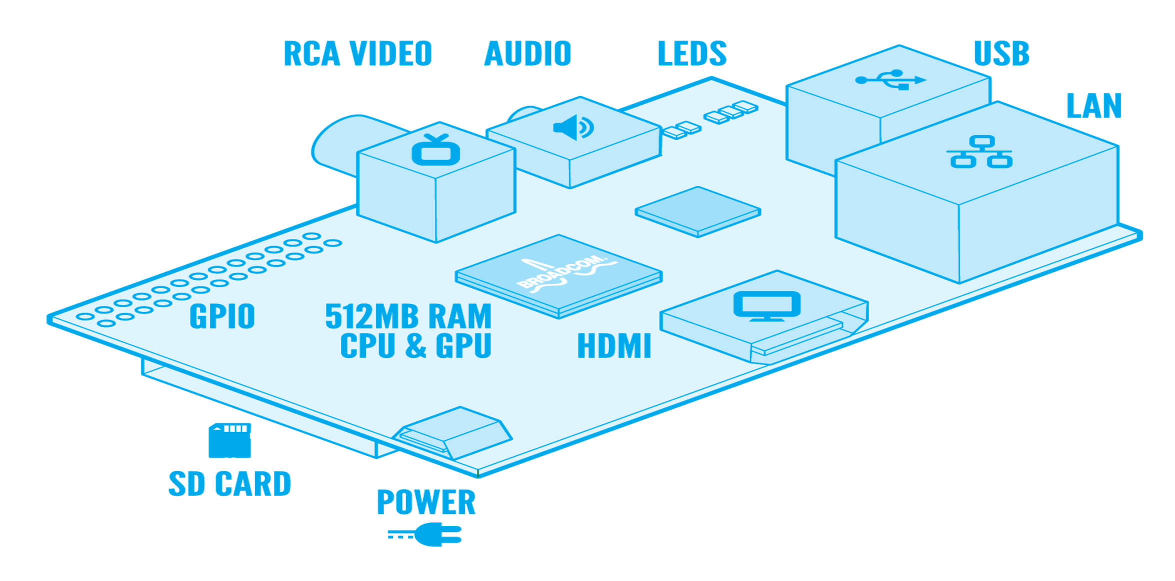
El modelo A de Raspberry es el modelo que se utilizó de prototipo final de la primera versión.

Este primer modelo dispone de la misma CPU que el B, pero no dispone de puerto Ethernet y además solo cuenta con un único puerto USB.

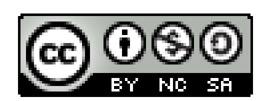
6.2 El modelo "B"

El modelo B de la Rasberry pi cuenta con 512 MB memoria, y dispone de dos puertos USB que pueden ser usados para conectar dispositivos periféricos.

Pero hay que recalcar que esos 2 conectores USB y el conector Ethernet, comparten el mismo Serial Bus.



Antonio Mónaco Osado



7 Conexiónes de una Raspberry

El dispositivo incluye conectores de expansión GPIO que pueden ser utilizados para la comunicación con otros dispositivos como módulos específicos de expansión, e incluso es compatible con Arduino, con lo que permite implementar un gran abanico de proyectos, que abarcan incluso la domótica.

Además con sus pines GND y +5V se pueden alimentar dispositivos como por ejemplo ventiladores, y posee dos puertos de expansión específicos donde podemos conectar periféricos para la Pi como su cámara HD.

Tanto en su versión con sensor infrarrojos, como la versión sin sensor.

RaspberryPi
SD CARD READER
Write your OS to SD Card
CAMERA EXPANSION
Using the CSI Connector
TFT TOUCH SCREEN
Using the DSI connector
HDMI-DVI-D ADAPTOR
MICRO USB POWER
5 V from mains
HDMI CABLE
HDMI/DVI
TV or Monitor
SD CARD
Class 2, 4, 6, 10* > 2 GB
LINUX OS
Debian Squeeze
Arch Linux Fedora
ARM
10/100 ETHERNET
Connect the Raspberry Pi to the internet
GPIO
CN13083
CN13088
STEREO AUDIO
3.5 mm jack
JTAG
Test/Debug
USB PERIPHERALS
Mouse / keyboard
POWERED USB HUB
Expand your ports
TV
Composite video to older TVs

Antonio Mónaco Osado

8 Sistemas operativos

En un principio Raspberry Pi solo disponía de Raspbian como único sistema operativo GNU/Linux y XBMC (Xbox Media Center) para conseguir algo parecido a una SmartTV conectándolo a cualquier TV de la que dispongamos, poco después apareció ArchLinux para ARM11 y se adaptó para la PI.

En los últimos años han aparecido muchos más sistemas operativos para este pequeño PC, por lo que he filtrado de entre ellos los cuatro que más óptimos me han parecido para mi proyecto, y finalmente me he decantado por el que más me convenía y el que más se adaptaba a lo que quería hacer.

8.1 FreeBSD

Es un avanzado sistema operativo para múltiples arquitecturas, entre ellas ARM (compatible con RPI).

FreeBSD es un derivado de BSD, la versión de UNIX desarrollada en la Universidad de California, Berkeley por el Computer Systems Research Group, y normalmente las distribuciónes de FreeBSD suelen destacar por su estabilidad.

8.2 Raspbian

Raspbian es pionero en Sistemas Operativos para Raspberry.

Es una adaptación de Debian 7 Wheezy para ARM, optimizada para Raspberry Pi y dispone de una gran comunidad detrás. Cada pocos días aparecen actualizaciónes, y su comunidad recuerda a la de Debian, pues procede de ella.

8.3 Arch Linux

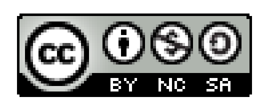
Arch Linux es una distribución de GNU/Linux que se basa en una filosofía similar a Slackware, donde no existían las herramientas de configuración automática.

Arch Linux exige absoluta maestría a la hora de configurar correctamente cualquier servicio o aplicación del sistema, además no lo he utilizado anteriormente, por lo tanto, por ese y otros motivos, queda descartado.

8.4 Kali Linux

Kali Linux, es una distribución pensada para pentesting muy famosa basada en Ubuntu, y posee múltiples herramientas para hacking ético y análisis informático forense.

Raspberry Kali Linux, es la distribución Kali para Raspberry optimizada para ARM, basada en Raspbian, esta opción es muy buena, pero al estar pensada desde su nacimiento para ser usada en PC lanzándose desde un LiveCD, trae demasiadas aplicaciones preinstaladas y configuradas, que no me hacen falta, y quizás puedan llegar incluso a entorpecer la implementación de mis servicios.



8.5 Por que Raspbian?

De entre los sistemas analizados anteriormente, me he decantado por Raspbian Wheezy, ya que está basado en Debian 7, con el que estoy bastante familiarizado.

Con este sistema operativo, voy a poder aplicar lo aprendido en clase, pero encontrándome con problemas que quizás en clase no han sido tratados, ya que es un procesador ARM.

9 Es posible hacer lo quiero hacer?

Antes de empezar con el proyecto, me he planteado la duda de si es viable o no.

Después de una gran búsqueda, me he dado cuenta de que todos los servidores que necesito, han sido implementados ya alguna vez sobre Raspberry Pi, y en concreto sobre Raspbian, pero no he encontrado referencias de que se hayan instalado todos simultáneamente en la misma raspberry.

La gran duda será si la pequeña Pi lo soportará o no, yo estoy seguro de que si, ya que si he probado servidores similares sobre algún Pentium 4 con 512 MB de memória RAM, la misma cantidad de memoria que posee el modelo B que voy a utilizar.
Lo único que me hace dudar es la arquitectura del procesador.

De todas formas hay que tener presente que raspberry se puede overclockear hasta conseguir 1GHz en su procesador y 500MHz en su memória RAM, incrementando en un 50% su poténcia, y también cabe recalcar que a finales de 2014 se lanza el esperado modelo “C”, que mantiene su arquitectura ARM, pero sube sus prestaciónes, entre otras pasando a tener 1 GB de ram, y mas prestaciónes en cuanto a CPU y GPU se refiere.

Hay que tener en cuenta que quizás al subir las frecuencias, se genere mas calor, y debamos refrigerar los chips que mas calor emiten, que són el chip SoC, el adaptador Ethernet, el chip de RAM, y el regulador de tensión que alimenta los componentes de la PI, como en este estudio se demuestra y quizás haya que refrigerarla de algún modo.

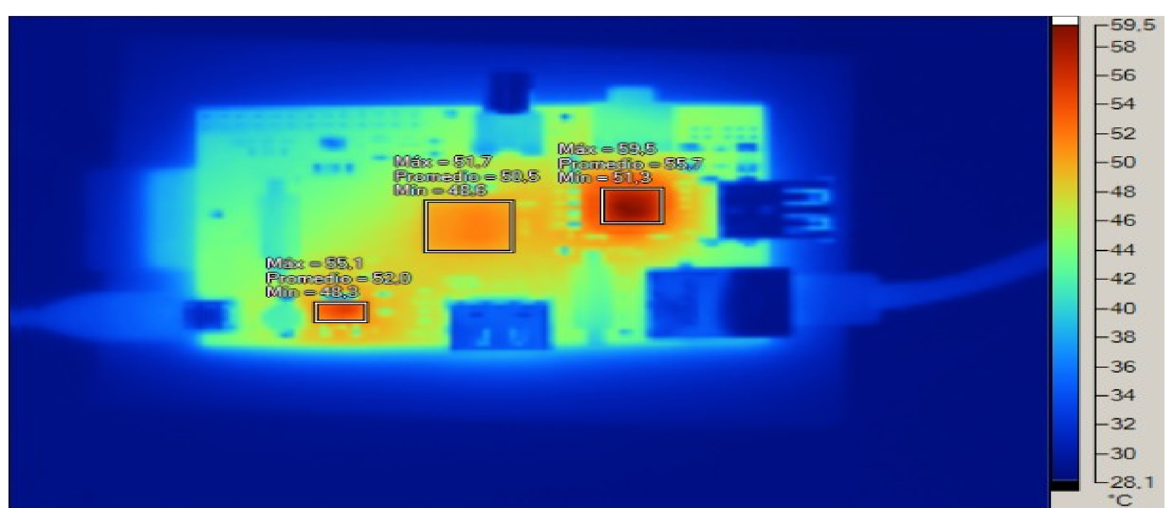
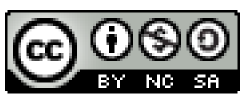


Foto extraida del enlace del estudio.





10 Una vez comprobada su viabilidad, que necesito?

10.1 Recursos materiales

Llegados a este punto, y viendo que es posible desarrollar este proyecto, ha llegado el momento de pensar en que recursos necesito para terminarlo.

Como recursos materiales necesitaré una Raspberry PI, al menos una tarjeta SD de 4 GB como mínimo, aunque no vendría mal tener dos o tres, para las copias de seguridad rutinarias, esta tarjeta será resistente a los golpes, al fuego, sumergible, etc..

Aunque finalmente me he decantado por una tarjeta de 16GB para ir cómodo, y un pendrive de 4GB del que disponia para utilizarlo de SWAP, aunque con uno de 2gb sería mas que suficiente. Después de varias pruebas he encontrado el archivo SWAP que raspberry utiliza por defecto, y he modificado el valor de 100, a 256MB, evitando así la saturación de la memória Swap en determinados momentos, y reduciendo el consumo eléctrico. Además 256MB es la cantidad correcta para 512MB de RAM, y no 4096.

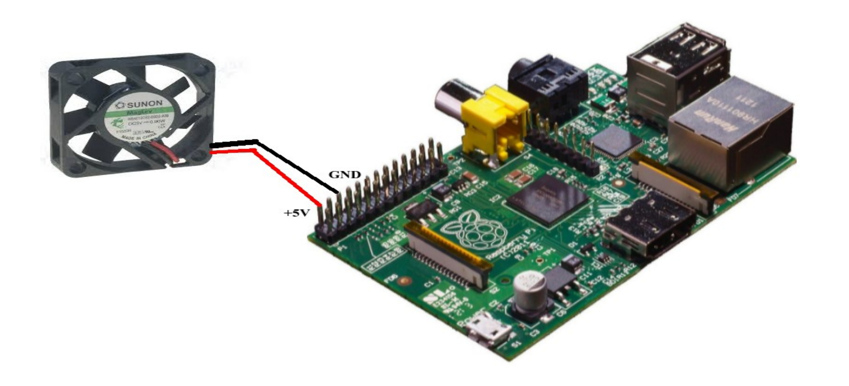
También voy a necesitar un par de cables UTP o STP y el Router Linksys para poder trabajar en una red local propia sin restricciones, porque toda la configuración se va a hacer mediante SSH o Webmin.

Finalmente he necesitado un kit de disipadores para la Pi, un ventilador de 5V marca SUNNON, la cámara con sensor infrarrojo, la caja con soporte accesorio para la cámara y un adaptador de puerto serie RS232 a USB

El presupuesto final es el siguiente:

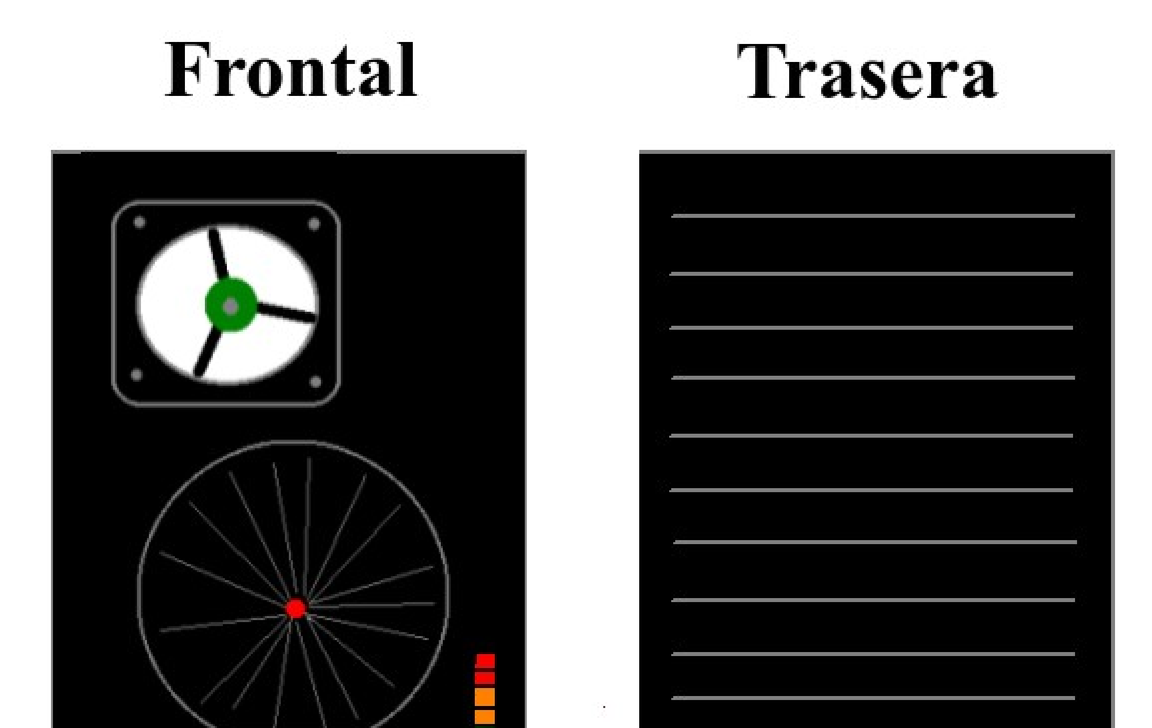
Producto	Precio	Cantidad	Precio total
Raspberry PI.	37,40 €	1	37,40 €
Caja Raspberry con soporte para cámara.	13,00 €	1	13,00 €
Cámara raspberry pi con sensor IR.	23,50 €	1	23,50 €
Adaptador USB-RS232.	15,00 €	1	15,00 €
Hub USB alimentación externa.	3,50 €	1	3,50 €
Cable Ethernet RJ45	2,50 €	3	7,50 €
Transformador microusb 5V 1,5A	5,00 €	1	5,00 €
Tarjeta SD 16 GB	14,70 €	1	14,70 €
Router Linksys WR54G	45,00 €	1	45,00 €
Ventilador 5V	6,00 €	1	6,00 €
Pegamento térmico	1,00 €	1	1,00 €
Kit disipadores RPI	7,00 €	1	7,00 €
Precio total	**171,60 €**		

Con el ventilador y los disipadores, la temperatura baja de 68º a 41 utilizando el 100% de los recursos de la Pi.



Antonio Mónaco Osado

Aunque esto es un prototipo, y si se va a fabricar a escala, recomiendo fabricar el siguiente diseño que he hecho con KolourPaint4 y Gimp, que dispone de agujero para ventilador de 5V, desplazando la cámara hacia abajo y mejorando la rejilla de ventilación en la parte trasera:



Además del ventilador he utilizado un kit de disipadores.



Antonio Mónaco Osado

10.2 Software necesario

Antes de instalar ningún paquete, actualizaremos el firmware de la Raspberry Pi, la lista de los repositorios, y las aplicaciones.

Esta acción debe hacerse automáticamente cada día, mas adelante lo implementaremos con crontab.

A continuación necesitaré implementar diferentes servicios para que Raspberry Pi actúe como servidor de impresión, escaneo, web, ssh, free rdp, dns, dhcp, compartición de archivos, cámara de videovigilancia, y finalmente como proxy transparente, ya que la carga de Joomla y Mediawiki era algo lenta en la Raspberry.

Para que Squid funcionara correctamente tuve que añadir una regla de IPTables que más adelante explicaré, así que como ya estaba escribi un completo archivo de reglas para el Firewall.

Una vez instalados, debemos conectar la Pi a una pantalla para ver que al incio del sistema todos cargan correctamente. Si es así, visualizaremos un OK de color verde junto al texto de carga del servicio.

10.2.1 Que paquetes debo instalar en mi servidor?

- Actualizar el firmware de la raspberry pi: rpi-update
- Servidor DHCP: isc-dhcp-server
- Servidor DNS: bind9 bind9-doc dnsutils
- Servidor de impresión: cups cups-pdf
- Servidor de escaneo: xinetd sane xsane libsane sane-utils
- Servidor SSH: openssh-server
- Servidor Free RDP: xrdp
- Servidor LAMP: apache2 php5 libapache2-mod-php5 php5-curl
- Servidor BBDD: mysql-server mysql-client-php5 mysql
- Servidor SMTP: postfix
- Servidor IMAP/POP: courier-imap courier-pop
- Servidor Proxy cache: squid squid-common
- Servidor FTP: vsftpd
- Cámara IP: motion-mmalcam
- En el router: Firmware DDWRT v24 sp1
- Aplicaciónes web: Joomla, PHPMyAdmin, MediaWiki, Webmin, Roundcube, OpenERP, Wordpress (Descargados de la web para obtener la última versión).



10.2.2 Configuración de los servicios

Para conseguir el objetivo final, deberé configurar los servicios que he instalado previamente.

Hay que configurar los siguientes aspectos:

- Servidor DHCP: Configurar el servidor para que asigne IP's de un rango con la mitad de IP's de la red y que asigne el servidor DNS, el gateway, el dominio, dirección broadcast y los default y max lease time en el archivo dhcpd.conf, tambien debemos especificar en ese mismo archivo de configuración las reservas para IP estáticas de la red.

- Servidor DNS: Crear una zona y agregar los hosts y alias necesarios. Añadir registro MX (servidor de correo) y NS (name server). Ahora hay que añadir la zona en el archivo named.conf.local y los forwarders en named.conf.options.

- Servidor de impresión: Configurar permisos en la interfaz web, y asignar ip al servidor, finalmente agregar impresora virtual con cups-pdf y guardar el archivo .ppd que contiene el driver de la impresora, ya que algunos sistemas antiguos como Ubuntu 10.04 no disponen del driver PDF.

Todo esto debe hacerse desde la web de administración de CUPS en el puerto 631, conectándonos desde el navegador web Midori de la Raspberry Pi accediendo por RDP.

- Servidor de escaneo: Asignar una ip al servidor en /etc/sane.d/saned.conf y configurar /etc/default/saned correctamente, debe quedar como el del anexo.

Hay que revisar la configuración del archivo /etc/inetd.conf y finalmente podemos comprobar si SANE está escuchando por su puerto por defecto que es el 6566 con ***netstat -penuta | grep sane***.

- Servidor SSH: Por cuestiones de seguridad debe revisarse que solo se permita SSH v.2, y debemos especificar los usuarios permitidos en la variable AllowUsers, y con MaxAuthTries definimos las veces que se puede fallar el password antes de bloquear el acceso al nodo que se ha confundido para evitar ataques por fuerza bruta, yo he definido uno solo.

En el archivo /etc/issue.net podemos escribir un mensaje de bienvenida que se mostrará al iniciar sesión, yo he escrito un mensaje de advertencia.

- Servidor Free RDP: El servidor RDP no debe configurarse, solo hay que saber que hace una especie de redirección por VNC, cosa a tener en cuenta para el script de IPTables a la hora de abrir puertos.

- Servidor LAMP: Configurar Apache2 i vigilar con los permisos en /var/www, siempre que haya problemas de permisos se puede arreglar con un simple ***chown www-data:www-data -R.***

He decidido activar el usermod para que cada usuario disponga de su directorio public_html en el que alojar sus páginas web, en el script anexo de instalación de LAMP se pueden ver las ordenes para activar dicho módulo.
- Servidor BBDD: Voy a gestionar las bases de datos con PHPMyadmin.

Para cada aplicación web que requiera de su propia base de datos, voy a crear un usuario propio que solo tenga permisos sobre su propia base de datos asociada.

- Servidor SMTP: Al instalar Postfix hay que seguir los pasos, escribiendo el dominio y la cuota de los buzones de los usuarios, yo he escogido una cuota de 512MB, ya que la veo razonable para el espacio del que dispone la raspberry, y las necesidades del correo de una empresa.

Lo único a tener en cuenta es el tipo de correo, como presuponemos que se utilizará con un dominio real, he escogido sitio de internet.

- Squid: En SQUID es necesario configurar el puerto por el que escucha, en este caso 3128, además podemos restringirlo a una IP con ese puerto, también hay que configurar squid para que todo lo relacionado con el proxy lo hagan el grupo y usuario proxy por motivos de seguridad.

Se pueden crear ACL's para decidir quien puede navegar, que paginas están permitidas, y configurar el tamaño de caché tanto en memoria como en disco, y hay que vigilar, y calibrar bien estos valores, ya que mas memoria no significa necesariamente mas rendimiento. Yo he escogido 16MB de caché en disco, y 128 en RAM.

Ya que con menos no cachea lo suficiente, y con más, tarda demasiado en buscar el contenido en cache y servirlo. El proxy caché aumentó bastante la fluidez del tráfico local, pero ralentizó tanto la conexión al exterior, tanto que se volvió casi inviable.
Había que encontrar alguna forma de solo cachear las webs locales.

Finalmente hay que escoger una opción entre las disponibles para configurar los clientes, yo he probado dos, y a continuación las voy a explicar, y el porque de mi elección que volvería a hacer funcional el proyecto.

Archivo de configuración automática proxy.pac:
Esta opción me ha gustado mucho, y firefox me avisa al cargar el archivo de configuración automática, pero implica que el cliente copie y pegue una dirección de configuración automática.

Además al configurar todo, aceleré el tráfico local gracias a la caché de squid, pero el tráfico hacia internet íva realmente lento.

Combinar IPTables con SQUID:

Finalmente me he decantado por esta segunda opción para solventar el problema que me surgió con la fluidez de las conexiones hacia internet.

La solución consta de una regla de IPTables, en la cadena de PREROUTING, que hace que todas las peticiones entrantes por el puerto 80, se redirijan al puerto 3128 que es el que usa SQUID.

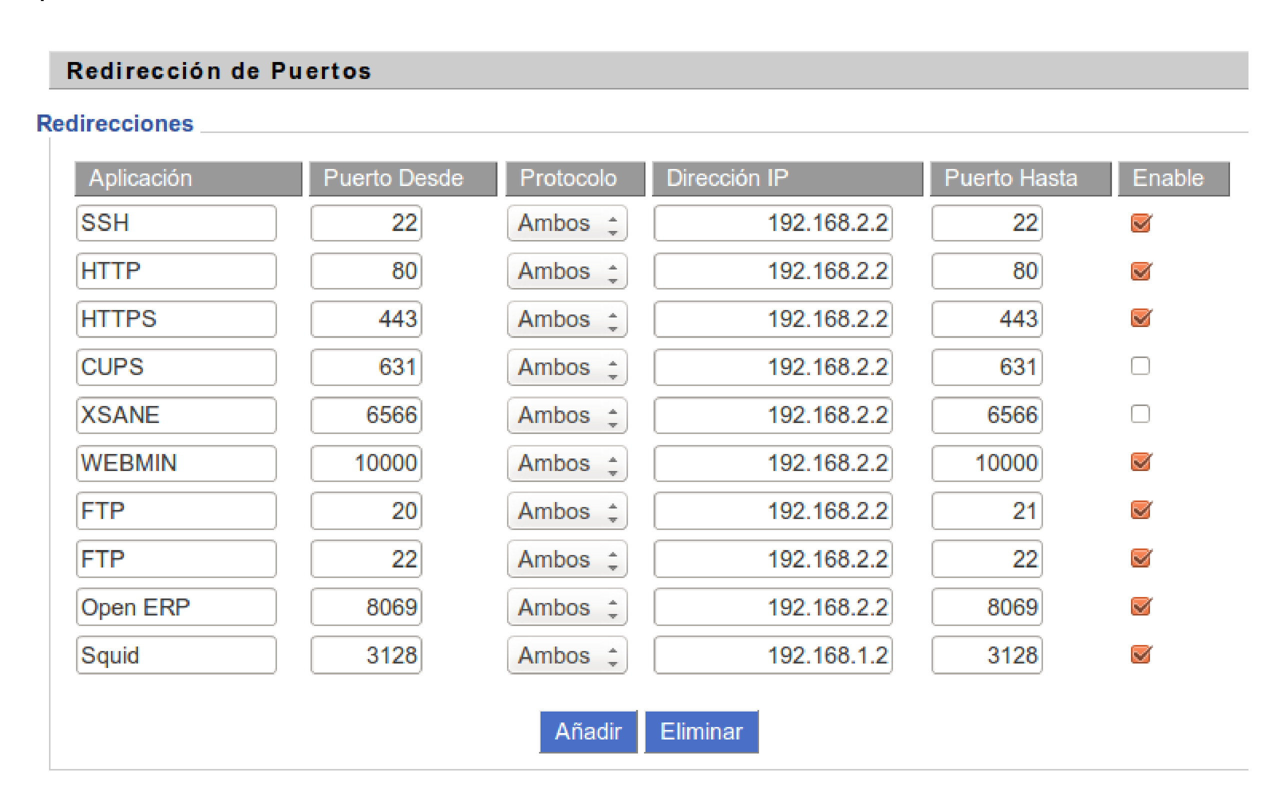
Esto me animó a completar el script y crear un Firewall para la pi que explicaré mas adelante.

- Servidor SAMBA: Debemos configurar smb.conf y establecer en el el dominio en WORKGROUP, y las carpetas compartidas con sus permisos, además descomentamos la línea del fichero de configuración que establece CUPS como el servidor de impresión local.

Y los servidores Courier IMAP y Courier POP los dejamos por defecto.

- En el router: Es necesario configurar el dominio WAN y el dominio LAN, el rango DHCP, el servidor DNS local y el del ISP, y hacer NAT de todas las peticiones entrantes por la interfaz WAN hacia la Raspberry en los puertos deseados, además deberemos configurar la red WIFI, la red local y epuerto WAN.

A continuación podemos ver una tabla de las redirecciones TCP y UDP (NAT) del router a la pi:



Redirección de Puertos

Redirecciones

Aplicación	Puerto Desde	Protocolo	Dirección IP	Puerto Hasta	Enable
SSH	22	Ambos	192.168.2.2	22	☑
HTTP	80	Ambos	192.168.2.2	80	☑
HTTPS	443	Ambos	192.168.2.2	443	☑
CUPS	631	Ambos	192.168.2.2	631	☐
XSANE	6566	Ambos	192.168.2.2	6566	☐
WEBMIN	10000	Ambos	192.168.2.2	10000	☑
FTP	20	Ambos	192.168.2.2	21	☑
FTP	22	Ambos	192.168.2.2	22	☑
Open ERP	8069	Ambos	192.168.2.2	8069	☑
Squid	3128	Ambos	192.168.1.2	3128	☑

Añadir Eliminar



10.2.3 Configuración de las aplicaciónes WEB

- MediaWiki: Es necesario instalar Imagemagik para que se puedan insertar miniaturas de imagenes en la wiki.

También hay que descargar LocalSettings.php tras la instalación en el navegador ycopiarlo en /var/www/mediawiki, además debemos tener mucho cuidado con el nombre de host de este archivo de configuración php, ya que si cambiamos de IP o dominio nuestra máquina, debemos reflejarlo en ese archivo, y si no no funcionará la wiki.

- PHPMyAdmin: Crear un usuario para mediawiki, otro para roundcube y otro para joomla, cada uno con su base de datos propia en la que disponen de todos los privilegios.

- Joomla: Crear una web publicitaria con una tienda donde se vendan los dos modelos de Raspberry modificadas.

- Wordpress: Solo lo he instalado, para que el cliente final disponga de la opción de crear su própio blog de empresa.

- Webmin: He creado un usuario solo con a los módulos que permitan configurar los servicios que he instalado, además he programado backups de /etc, /var y /home que se ejecutan cada día a las 22:40h y comprueban el backup simulando una restauración, y uno semanal de todo el sistema de archivos.

Al finalizar los backups, se envía un log al e-mail del administrador.

- PHPSane: Retocar el archivo de configuración para permitir escaneo desde localhost.

Finalmente PHPSane no ha funcionado, pero he encontrado una aplicación cliente Sane para cada sistema operativo, así que he seguido adelante.

- RpiMonitor: He encontrado esta útil aplicación web en RPIStore, una especie de GooglePlay para Raspberry PI.

Permite monitorizar CPU, memoria, uso de disco, temperatura, etc... remotamente, y almacena la información con la que crea gráficos que podemos visualizar.

- Open ERP: Solo la he instalado para el cliente final.

- Página de inicio: Archivo HTML con su CSS para el diseño.

Está maquetado con <DIV> y utilizo la etiqueta <MARQUEE> para el texto desplazable, además con la etiqueta <IFRAME> he empotrado Rpi Monitor, y finalmente he cogido esta página a modo de plantilla para crear la web picam.html, para la que he añadido algunas clases y estilos adicionales al CSS.

10.2.4 Comandos/programas UNIX GNU/Linux necesarios

Dd: lo utilizaré para hacer imágenes de la SD de raspberry en desarrollo.

SCP: lo utilizaré para los backups completos de la SD de raspberry en producción.

Mkswap: lo he utilizado para crear una swap con un pendrive, ya que de serie vienen 100MB y a veces se saturaba.

Fdisk: Lo he usado para ver la UUID del disco SWAP.

Traceroute: Lo he usado para trazar las rutas que seguían mis paquetes.

Ssh-copy-id: Lo he usado para copiar las claves públicas al servidor para acceder sin necesidad de contraseña y automatizar los backups

ssh-keygen: Lo he usado para crear el par de claves RSA para SSH.

GPG: Lo he usado para crear un par de claves para cifrar y descifrar los archivos, las claves són de 4096 bits.

Además he editado la clave pública gpg de mi portátil firmandola con la clave de la raspberry para automatizar los backups cifrados, ya que me pedía que verificara cada vez que la clave era fiable.

Tar: Lo he utilizado para comprimir los backups que he programado con extensión .tar.gz

Mysqldump: Lo he utilizado para respaldar las bases de datos utilizandolo en scripts en convinación con tar, gpg y cron.

Sh: He implementado diferentes scripts en sh.

Fsck.fat: Lo he utilizado para corregir la corrupción del sistema de archivos mmcblk0p1 (partición /boot de la Raspberry).

10.2.5 Programario de gestión de proyectos

Para llevar a cabo este trabajo, necesitaré algunos programas para poder gestionar el proyecto como GNU Planner para hacer el diagrama de Gantt, usb-imagewritter para grabar imagenes en la SD, block de notas basket para poder llevar a cabo la investigación en una interfaz muy potente y versátil que permite después exportar las cestas de apuntes a HTML para compartir posteriormente cientos o miles de archivos de texto plano con un solo click y Dia, GIMP y Kolourpaint4 para hacer los diagramas,

10.3 Instalar Raspbian

Para instalar Raspbian, primero debemos descargar USB Imagewritter, y instalarlo ejecutando

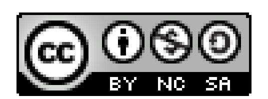
```
sh install.sh
```

Una vez instalado, descargaremos Raspbian de la página oficial, y quemaremos la imagen en una tarjeta SD de 4 GB mínimo.



Tuve un contratiempo con mi lector de tarjetas SD, ya que se monta en /dev/mmcblk0, y USB-Imagewritter no localizaba ningún dispositivo puntero, es decir, ninguna tarjeta SD o pendrive conectado, ya que apunta a los dispositivos /dev/sd*.

En este punto había dos opciones, y nos me he decantado por la más sencilla, en vez de editar los archivos de texto que componen el programa, en busca de la variable que contenga el punto de montaje al que ha de ir a mirar el software, he buscado una forma de imitar USB-Imagewritter, y esto lo he hecho con el comando ***dd.***



Adicionalmente, he reducido el tiempo de backup a aproximadamente 5 minutos y medio con una SD HC de 4GB, indicándole a dd que debe transferir en bloques de 1MB.

Con la tarjeta de 16GB tarda aproximadamente 25 minutos.

Esto no afecta después al sistema, ya que estos bloques, solo son los “trocitos” en los que dd descompone la información para transferirla en pequeños paquetes de 1MB, es decir, el tamaño de bloque de Raspbian una vez volcado el backup a la SD, es el que tiene por defecto.

El backup se compone de las siguientes dos líneas:

Antes de realizar el backup, debemos desmontar la SD, que se compone de /dev/mmcblk0p1 (**/boot**) y /dev/mmcblk0p2 (**/**).

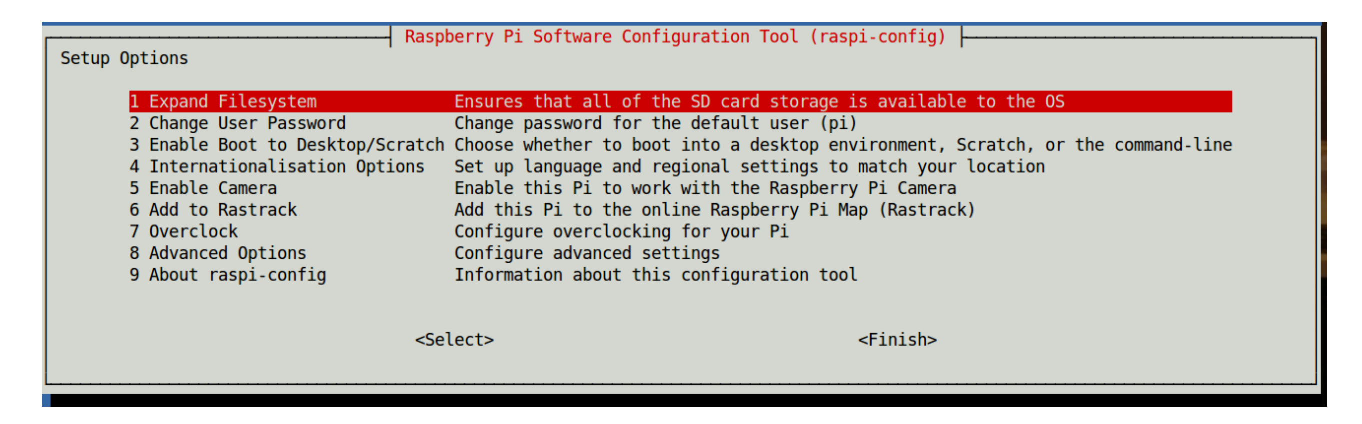
Hay que desmontarlas por separado, ya que mmcblk0 no es una partición.

Finalmente copiamos con dd. En “if= ” pondremos el directorio que queremos copiar, y en “of= “ la ruta del destino del archivo de respaldo, esto nos va a servir de mucho, ya que prescindiremos de imagewritter, a cambio de este comando que además de hacer lo mismo, volcar una imagen, nos permite hacer la inversa en este caso, y realizar un backup.

He escogido .img, ya que todas las distribuciones para RPI se facilitan en ese formato.

```
sudo umount /dev/mmcblk0p1 && sudo umount /dev/mmcblk0p2
sudo dd if=/dev/mmcblk0 of=/media/tmo/Data/raspberrypi.img bs=1M
```

Al entrar por primera vez, encontraremos rpi-config, una pantalla de configuración que es lo mas parecido a una BIOS que posee la Raspberry Pi.



Este menú nos permite entre muchas otras opciones escoger franja horaria, distribución de teclado, activar/desactivar SSH o la PiCam, overclockear CPU y RAM, o actualizar el Firmware.

Finalmente si en algún momento quisiéramos acceder a ese menú simplemente deberemos teclear raspi-config en la terminal de raspbian.

Haciendo pruebas me di cuenta de que las opciones como el overclock y el overvolt, raspi-config las configura en /boot/config.txt, así que finalmente configuré estos aspectos a mano en dicho archivo de configuración con un resultado satisfactorio.

Los valores elegidos fueron 950MHz para el procesador, 250MHz para la RAM y un valor de 6 para el overvolt, para ganar rendimiento sin forzar al máximo la Pi, alargando su vida un tiempo más y evitando así errores de ejecución y/o funcionamiento, aunque la garantía cubre el overclock de la Pi hasta 1GHz.

Se debe tener en cuenta que el overclock se aplica aunque veamos que la CPU sigue a 700MHz, ya que cuando es necesario, el valor sube a 950 para satisfacer la demanda de los servicios y las aplicaciones, volviendo a 700MHz cuando no es necesario más.

Esto se debe a que la CPU trabaja en scaling governor ondemand.

10.3.1 Actualizar el Firmware de Raspberry Pi

Aunque raspi-config nos permite actualizar el firmware con el comando ***rpi-update***, he desarrollado un script que descarga la última versión del firmware de la página oficial con el comando ***wget***, y acto seguido lo instala con el mismo comando.

Finalmente deberemos reiniciar para aplicar los cambios, y llegados a este punto podemos actualizar los repositorios y el software con ***apt-get update && apt-get upgrade***.

Al acabar este apartado añadí el script de actualización del firmware a crontab, consiguiendo así que cada día mientras la empresa está cerrada por la noche, se actualice automáticamente el firmware de Raspberry, sin influir en el trabajo de los empleados, ya que esto sucede en horario de descanso, concretamente a las 23:00 cada día, y así implementaré las demás tareas de mantenimiento.

El script de actualización de las aplicaciones y los repositorios lo he copiado en /etc/cron.hourly, así el software busca actualizaciones cada hora, y si las hay las instala automáticamente.

Al actualizar el Firmware, aparece un error que dice que falta la carpeta de prueba I2C, y es un error que se puede ignorar, ya que es un bug que ha quedado en el firmware después de las pruebas finales de la tarjeta de sonido para la Pi, como puede verse en este enlace.

Encontramos también un error de corrupción del sistema de archivos mmcblk0p1, es decir la partición /boot de Raspberry, parece que es algo habitual, y en un foro en inglés encontré la siguiente solución, pues fsck no funciona con esta partición, una vez ejecutada esta orden, el problema queda resuelto.

Fsck.fat -trawl /dev/mmcblkp01

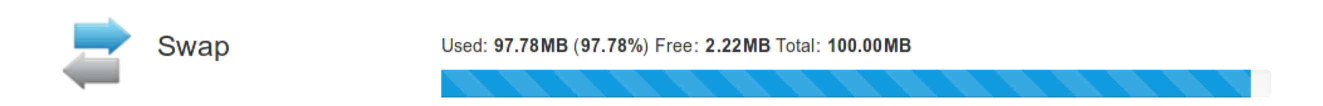
Lógico, pues la partición boot, no es ext4 como /, sinó que es FAT32.

Fsck.fat -trawl, según su manpage, realiza los siguientes cambios en el sistema de archivos:

La **"t"** marca los clústers no legibles como dañados, la **"r"** repara el sistema de archivos, la **"a"** automatiza la tarea para que no se requiera interacción con el usuario, la **"w"** escribe los cambios inmediatamente en el disco y la **"l"** lista el path de los archivos procesados.

10.3.2 Ampliar SWAP

Una vez probado todo, la SWAP de 100MB aveces se saturaba, y la Raspberry se ralentizaba.



Primero decidí acoplar un pendrive de 4GB y con mkswap utilizarlo como swap para solucionarlo, finalmente con swapon he probado que funcionaba, y para acabar la he añadido a /etc/fstab para que se monte automáticamente al arrancar el sistema.

```
proc            /proc           proc    defaults          0       0
/dev/mmcblk0p1  /boot           vfat    defaults          0       2
/dev/mmcblk0p2  /               ext4    defaults,noatime  0       1
# a swapfile is not a swap partition, so no using swapon|off from here on, use  dphys-swapfile swap[on|off]  for that
```

```
pi@raspberrypi ~ $ sudo blkid /dev/sda
/dev/sda: UUID="5e03122f-8474-43c2-ab1f-eda9dd365f7a" TYPE="swap"
pi@raspberrypi ~ $
```

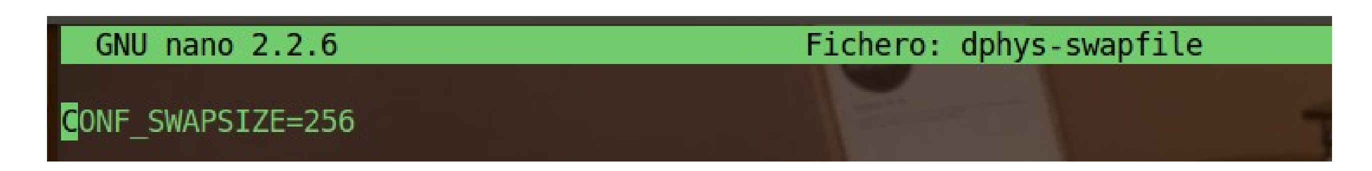
```
  GNU nano 2.2.6                           Fichero: /etc/fstab

proc            /proc           proc    defaults          0       0
/dev/mmcblk0p1  /boot           vfat    defaults          0       2
/dev/mmcblk0p2  /               ext4    defaults,noatime  0       1
UUID=5e03122f-8474-43c2-ab1f-eda9dd365f7a none  swap    sw      0       0
```

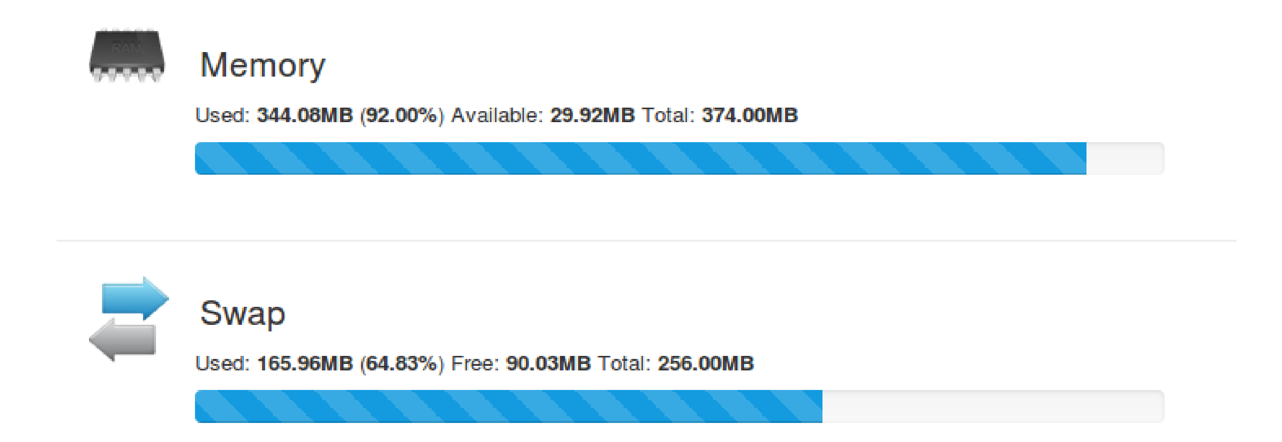
Con mount -a se empieza a usar la nueva SWAP, pero debemos reiniciar rpimonitor con sudo service restart, ya que muestra mal la capacidad:



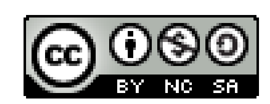
Al realizar mas pruebas, encontré el archivo de Swap de raspberry, /etc/dphys-swapfile y modifique allí el valor de 100MB por 256MB.



Consiguiendo el mismo rendimiento, menor consumo eléctrico, y un valor de SWAP mas ajustado a los 512MB de RAM que trae la Raspberry.



Antonio Mónaco Osado



11 Implementación

A estas alturas, ya vamos a empezar la implementación de los servicios y aplicaciones necesarios/as que hemos visto anteriormente.

El primer paso de este apartado será proceder a la instalación de los mismos.

11.1 Instalación del servidor RPI

Para proceder a la instalación, y entregar el trabajo a tiempo, he decidido guardar en archivos ejecutables todo lo que haga y funcione correctamente, para posteriormente no tener que repetir todas las tareas sobre Raspbian, ya que en un principio esperaba la compra por parte del instituto de una Raspberry.

11.1.1 Aplicaciónes

Para instalar las aplicaciones, he anotado los paquetes necesarios en un archivo ejecutable con extensión sh (un script) para posteriormente automatizar la instalación de las mismas en la Raspberry.

Este archivo simplemente usa el siguiente comando con los paquetes como argumento.

```
Sudo apt-get install -y aplicacion1 aplicacion2
```

Con el se instalan los siguientes paquetes:

perl libnet-ssleay-perl openssl libauthen-pam-perl libpam-runtime libio-pty-perl apt-show-versions python unzip imagemagick locate ckermit aptitude rkhunter chkrootkit clamav clamtk mcrypt ntp filelight

11.1.2 Servicios

Para instalar los servicios y sus dependencias, he instalado una a una anotando los pasos, para finalmente tambien redactar scripts en sh para no tener que repetir la tarea sobre Raspberry.

La mayoría de estos scripts solo instalan los servicios, pero algunos también modifican permisos, propietarios y grupos de directorios, o activan módulos para esos servicios.

Algunos ejemplos podrían ser:

```
#Instalamos LAMP y sus dependencias
sudo aptitude install -y apache2 php5 libapache2-mod-php5 php5-curl
sudo aptitude install -y mysql-server mysql-client php5-mysql

#Forzamos a Apache2 a recargarse
sudo /etc/init.d/apache2 force-reload

#Damos permisos a www-data sobre /var/www
sudo chmod -R 755 /var/www/
sudo chown -R www-data /var/www
sudo chgrp -R www-data /var/www

#Activamos los directorios personales de los usuarios (public_html)
sudo a2enmod userdir
sudo /etc/init.d/apache2 force-reload
sudo mkdir /etc/skel/public_html
sudo chmod -R 755 /etc/skel/public_html
```

Al final los he ejecutado sobre Raspbian y el único problema que he encontrado ha sido que al instalar Apache2, tanto el grupo www-data como el usuario con el mismo nombre, y que pertenece a ese grupo no existían.

Lo he solucionado añadiendo al script de instalación de LAMP las siguientes líneas.

```
sudo groupadd -f -g33 www-data
sudo useradd -g www-data www-data
```

También he necesitado instalar el módulo cups-pdf para CUPS, para poder crear una impresora virtual, ya que inicialmente planteé usar PDF Creator porque era el que conocía, pero es para Windows únicamente.

Investigando un poco me di cuenta de que la impresión en PDF por defecto que trae Ubuntu, se lleva a cabo gracias al cliente y el servidor CUPS y el paquete cups-pdf que dota al sistema de una impresora virtual en PDF, y este módulo también se puede instalar en un servidor para compartir esa impresora virtual a los clientes de la red local.

Finalmente he decidido añadir Samba, ya que tanto Linux como Windows implementan estos protocolos por defecto en sus variantes, y sin necesitar de software adicional, el usuario puede descargar los PDF que imprime con cups-pdf (Impresora PDF) de la carpeta compartida por SAMBA.

Si se han imprimido por un usuario registrado en la Pi, estos se almacenan en la home de ese usuario por defecto en el subdirectorio Impresora_PDF.

Si la impresión es anónima, irán a la carpeta Impresora_PDF/ANONYMOUS, disponible para todos vía SAMBA.

Los usuarios de GNU/Linux al acceder a la red dispondrán automáticamente de las impresoras locales, gracias a CUPS, pero los usuarios de Windows deben instalar sus drivers, accesibles vía SAMBA en el servidor.

Así gracias a la implementación de SAMBA podemos compartir impresoras con los usuarios de nuestra empresa que utilizen Windows, permitiéndoles instalarla con un solo doble-click accediendo por SAMBA al directorio “Shared Printers”

Los usuários Linux, Mac y Android detectan automáticamente las impresoras.



11.1.3 Aplicaciónes WEB

Las aplicaciónes Web las he instalado descomprimiéndolas en /var/www.

Tras descomprimirlas, volvemos a hacer propietario de www y sus sub-carpetas a www-data con la orden ***chown www-data:www-data -R /var/www*** accediendo a ellas con el navegador web y siguiendo los pasos de la instalación.

Para cada aplicación web que requiera una base de datos, hay que crear un usuario con su BBDD asociada, para mayor seguridad desde PhPMyAdmin, que es la primera APPWEB que debemos instalar.

Algunas aplicaciónes web como OpenERP són paquetes .deb que deben instalarse con ***dpkg -i nombrepaquete***.

Otras aplicaciónes web como MediaWiki disponen de un archivo de configuración, en este caso llamado LocalSettings.php y hay que tener especial cuidado con los parámetros que este archivo contiene, sobretodo el hostname, ya que si cambiamos de hostname o IP y no lo reflejamos en ese archivo, el servidor web no nos dará acceso a la pagina.

Además la mayoría de estos archivos se crean durante la configuración de las aplicaciones Web, y si no están establecidos los permisos necesarios, deberemos copiar el texto y crear nosotros el archivo a mano con algún editor de textos como **nano** y pegar el contenido en su interior, esto por ejemplo nos puede pasar también con phpmyadmin, mediawiki y joomla.

Estos archivos se copian a la carpeta raíz de la aplicación web, así el de MediaWiki hay que copiarlo en **/var/www/mediawiki**.




11.1.4 Necesidad de instalar el servicio SQUID

Con tantas aplicaciones Web, el servidor Apache2 respondía un poco lento, así que surgió la necesidad de instalar un proxy transparente, así que decidí implementar SQUID, ya que lo hemos aprendido a instalar y configurar en clase, así que en el script de instalación de servicios añadí los paquetes de SQUID.

Solo debemos configurar los valores de memoria en disco y en RAM, para los que he escogido 16 MB de memoria en disco y 128 en RAM.

También hay que configurar el puerto por el que escucha squid, yo he dejado el puerto por defecto 3128, pero he indicado que debe escuchar solo por la IP de eth0, además he definido que será un proxy transparente, los DNS, y el e-mail del admin del servidor.

Para que SQUID se auto configurase en los clientes, intenté hacerlo mediante un archivo proxy.pac alojado en /var/www, y los navegadores lo cargaban, pero no funcionaba óptimamente, ya que también pasaba por el proxy el tráfico que proviene y va hacia internet, lo que hacia que fuera muy lento.

Finalmente, tras pensar algunas horas, decidí añadir una regla IPTables en la cadena PREROUTING, que redirige el trafico que entra por eth0 a través del puerto 80, hacia el puerto 3128, y tras comprobar los logs de squid access_log, cache_log y store_log, pude comprobar que todo funcionaba correctamente, y pude comprobar que el ti del servidempo de respuesta del servidor Apache2 había bajado bastante.

Además verifiqué que esto funcionaba correctamente viendo que al editar MediaWiki, la IP que aparecía como mía era la de la PI, y yo estaba editándola desde mi portátil.

11.1.5 Camara de vigilancia con PiCam y Motion

En clase, Pau sugirió provar Motion a unos compañeros, y finalmente a mi.

Esa idea me atrajo mucho, y empecé a buscar sobre ello en internet, encontré varias páginas, y pude ver que Raspberry no es compatible con motion, pero si con **motion-mmalcam**, que es una modificación del binario de motion, y de su archivo de configuración que circula por la red.

Tras leer bastantes blogs, foros y webs oficiales, dí con este enlace en el que podemos descargar el archivo de configuración y el binario desde un dropbox, y una vez descargado el archivo, configuramos motion-mmalcam.conf como deseemos.

Los valores a tener en cuenta son los formatos en que almacena las imágenes, la frecuencia, el directorio donde se guardan, el nombre de las mismas y los puertos control y stream de motion.

Finalmente las paletas de colores, pero la que viene por defecto es la que debemos usar, ya que motion ofrece 8 por defecto, pero en esta modificación disponemos de 17, y la que viene por defecto es la adecuada para la PiCam, también podemos editar la calidad de las imágenes guardadas en disco, y la calidad del stream, el brillo, las frames por segundo, y el texto que veremos en la web de stream.

Yo he establecido el puerto 888 para stream y el 8088 para control, también he escogido el directorio /media/Camara_IP para guardar las imágenes, este directorio está compartido por SAMBA, y solo pueden acceder los usuarios root y vigilante.

La web de stream de Motion, está restringida a la IP del vigilante por IPTABLES.

Para ejecutar motion-mmalcam es necesario ubicarse en el directorio donde dispongamos del binario y el archivo de configuración y ejecutar la siguiente orden en la terminal:

./motion-mmal -n -c motion-mmalcam.conf

Esta orden ejecuta el binario motion-mmal, -n lo hace en modo no-daemon, y -c motion-mmalcam.conf carga el archivo de configuración modificado.

Quise añadir autenticación por password, pero Motion utiliza una base de datos que hay que crear enteramente a mano, con valores y campos que no hemos visto en clase, y no me íva a dar tiempo, ya que los tutoriales que encontré estaban basados en una versión antigua de mysql, y los comandos eran distintos, así que me decanté por el módulo Apache access_control, pero no conseguí implementarlo en motion, ya que no trabaja directamente desde /var/www, ni utiliza el puerto de Apache2.

Las imágenes de la cámara se guardan en .jpg y .avi simultáneamente y graban una imagen por segundo, cada hora los archivos .jpg se mueven al directorio /media/Camara_IP/Fotos y los vídeos a /media/Camara_IP/Videos, gracias a anacron y un script que se aloja en /etc/cron.hourly.

Cada 15 días las imágenes se purgan de la Pi, y se avisa al Vigilante y al Admin por e-mail, y debe quedar una copia de las mismas en el servidor de Backups, que definiremos mas adelante en el apartado backups.

Al funcionar bien y ser útil, decidí añadirlo al panel central, creando un panel aparte para las cámaras, ya que se podrían gestionar diferentes cámaras alojadas en diferentes raspberry's, pudiendo también dividir los servicios entre ellas para ganar fluidez y estabilidad.

Pero como solo dispongo de una PiCam y una Raspberry Pi, taché todas las cámaras menos una del panel, para dejar claro que disponibles tenemos una de momento, dejando las imágenes sin tachar disponibles en el directorio /var/www/rpsolutions/picam por si algún cliente requiere mas cámaras.

Además tanto en el panel como en la cámara, se puede poner el nombre del sitio que se está monitorizando.

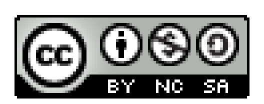


Panel de gestión de las cámaras de videovigilancia.



Cámara de videovigilancia.

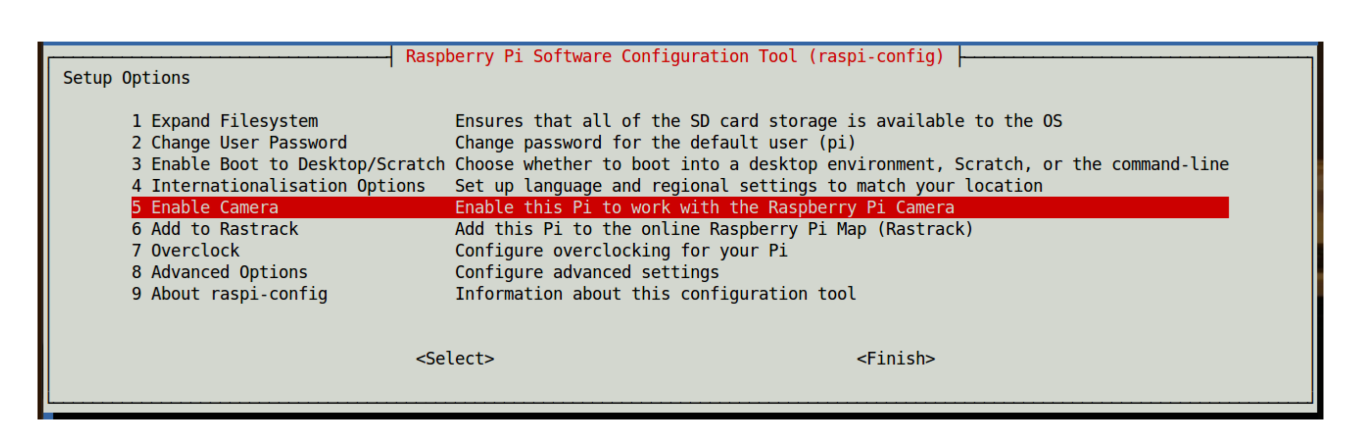
Antonio Mónaco Osado



Para conectar la PiCam a la Raspberry debemos hacerlo como se ve en la siguiente imagen:



Y debemos activarla en raspi-config



Finalmente para empezar a grabar en cada inicio del sistema, he creado un script, y he añadido una línea en /etc/rc.local para que se ejecute cada inicio del sistema, con esto también evitamos que conectando un teclado se pueda tener acceso a la pi, restringiendo el acceso únicamente vía SSH.

Antonio Mónaco Osado

11.1.6 Firewall con IPTABLES

Al configurar SQUID y utilizar IPTables para resolver mi problema del proxy transparente y la velocidad, decidí que debía escribir un script completo para la configuración del Firewall.

Así que he escogido política por defecto permisiva, abriendo los puertos deseados a mi LAN y después cerrándoselos a los demás, y me he ayudado con variables para agilizar el trabajo. El localhost lo he dejado abierto para conexiones locales.

Intenté utilizar preload y postload para eth0 en /etc/network/interfaces, para que al iniciar eth0 se cargara mi script, y al desactivar la interfaz hiciera un flush antes de reiniciar, pero como no me funcionó, añadí una línea al script /etc/rc.local que ejecuta el script en sh con las reglas de IPTables cada vez que inicia el sistema, con lo que no quedé muy contento por ser una chapucilla, ya que un firewall es algo muy delicado.

Finalmente copié el script a /etc/init.d y le añadí una cabecera como las de los scripts alojados en esa ruta, y ejecuté update-rc.d iptables start, con lo que conseguí que se crearan links en los directorios /etc/rc*.d, con lo que se ejecuta automáticamente en los runlevels escogidos.

En este caso en el 1, 2, 3, 4 y 5. En el 0 y el 6 se detiene.

A continuación puede verse una tabla de los puertos necesarios para este proyecto y sus servicios.

Esta tabla me ha ayudado a la hora de escribir el script que adjunto en los anexos:

Puerto	Servicio	Puerto	Servicio	Puerto	Servicio	Puerto	Servicio	Puerto	Servicio
20	FTP	80	HHTP	143	IMAP	993	IMAPS	6566	SANE
21	FTP	110	POP	443	HTTPS	995	POPS	8069	OPENERP
22	SSH	137	SAMBA	445	SMTPS	3128	SQUID	8088	MOTION – Admin port
25	SMTP	138	SAMBA	631	CUPS	3389	FREERDP	8888	RPIMONITOR
53	DNS	139	SAMBA	888	MOTION – Stream port	5900	VNC	10000	Webmin

11.1.7 Antivirus

Las empresas suelen ser foco de virus y todo tipo de malware, por lo que debemos instalar un antivirus y programarlo para que escanee el servidor periódicamente gracias a crontab.

Para ello he instalado ClamAV, y he creado un script que Cron ejecuta todos los días a las 04:30 cada día, para que no interfiera con el trabajo de los usuarios, ya que ClamAV consume muchísimos recursos.

Este script analiza todo el sistema de archivos de la pi, excluyendo los directorios /sys y /proc.

11.1.8 Antirootkits

Como conozco un par de buenos antirootkits, chkrootkit y rkhunter, voy a implementarlos en este servidor y voy a crear un script para cada uno de ellos, que Cron ejecutará periódicamente a las 00:10 y 00:25 respectivamente.

11.1.9 Cópias de seguridad

Las cópias de seguridad que he configurado, utilizan Webmin o scripts asociados a Cron.

Webmin utiliza SSH para hacer los backups, y envía un log por email al admin una vez terminada la tarea.

Yo he configurado los siguientes backups en Webmin:

Scheduled Backups

Seleccionar todo. | Invertir selección.

Directorio a respaldar	Sistema de archivos	Respaldar a	¿Programado?	Horas programadas para respaldar	Action
/etc	TAR	192.168.1.10:/home/tmo/Escritorio/Raspberry/diaria /etc.tar.gz	Si	After /var	Backup..
/var	TAR	192.168.1.10:/home/tmo/Escritorio/Raspberry/diaria /var.tar.gz	Si	Todos los dias a las 1:00	Backup..
/home	TAR	192.168.1.10:/home/tmo/Escritorio/Raspberry/diaria /home.tar.gz	Si	After /etc	Backup..
/	TAR	192.168.1.10:/home/tmo/Escritorio/Raspberry/semanal /Raspbian.tar.gz	Si	Semanalmente (el Domingo)	Backup..

El backup de / semanal, excluye los directorios proc, lost+found, mnt y sys.

También he escrito algunos scripts para otros backups, que mueven los archivos a respaldar a un directorio temporal que primero se crea, donde se comprimen, y en algunos casos los cifran, le añaden la fecha al nombre, y finalmente los envían al servidor de backups por scp, y borran el directorio temporal.

Para que no pidiera confirmación a la hora de realizar el backup. y fuera así automático, después de importar la clave pública gpg del servidor de backups tuve que firmarla con la clave privada gpg de la Raspberry.

La clave gpg se genera con la orden ***gpg --gen-key*** y yo la he creado RSA de 4096 bits, los máximos permitidos, y se firma con la orden ***gpg –edit-key***, seguido del comando ***sign,*** finalmente escribimos los cambios con la opción ***write***.

Para descifrar el backup debe hacerse desde el servidor de backups, y tenemos que usar la orden ***gpg -d Nombrebackup.gpg*** e introducir el password de la clave pública del usuario tmo en el servidor de backups, en este caso mi portátil.

Además la clave privada para acceder por SSH debe ser sin passphrase, y se genera con la orden ***ssh-keygen***.

Estos scripts los he utilizado junto a cron y anacron para respaldar los siguientes datos en las siguientes frecuencias:

Datos	**Hora o día en que se respaldan**
Imagenes de la cámara de vigilancia	00:45 cada día, gracias a /etc/crontab
Base de datos MySQL completa	23:55 cada día, gracias a /etc/crontab
Base de datos joomla de MySQL	Cada hora, el script está ubicado en /etc/cron.hourly
Base de datos roundcube de MySQL	Cada hora, el script está ubicado en /etc/cron.hourly
Base de datos mediawiki de MySQL	Cada hora, el script está ubicado en /etc/cron.hourly

Todas las copias de seguridad se han programado en horas de descanso, en las que la oficina está vacía, para que no se sature la red en horario laboral.



11.1.10 Otras tareas programadas con cron

He creado algunos scripts más para actualizar el firmware de la pi, pasar los antirootkits y el antivirus, hacer un purgado de la caché de squid, y vaciar el registro de las cámaras de vigilancia, estos scripts están en los anexos.

Tarea	Hora o día en que se realiza
Actualizar el firmware de la Pi	23:00 cada día, gracias a /etc/crontab
Ejecutar analizador de rootkits Chkrootkit	00:10 cada día, gracias a /etc/crontab
Ejecutar analizador de rootkits RKHunter	00:25 cada día, gracias a /etc/crontab
Ejecutar antivirus ClamAV	04:00 cada día, gracias a /etc/crontab
Purgado de la caché de Squid	07:30 cada día, gracias a /etc/crontab
Vaciamos el archivo de la cámara de videovigilancia	Día 15 de cada més a las 03:00, gracias a /etc/crontab
Copia de archivos de configuración modificados (toda aplicación configurada a mano).	Cada hora, el script está ubicado en /etc/cron.hourly

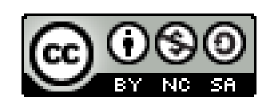
Todas las tareas se han programado en horas de descanso, en las que la oficina está vacía, para que no se sature la red en horario laboral.

11.1.11 Envío de Logs por correo

Para enviar logs de las tareas programadas con Cron y Anacron, he utilizado mailutils para enviar emails desde la terminal, el usuario indicado en cron debe tener Maildir en su home para que esto sea posible, y la síntaxis es la siguiente:

cat texto.txt | mail -s "Asunto del email" admin@rpsolutions.com.es

Dentro de texto.txt redirecciono el mensaje a enviar mientras se van ejecutando los scripts.



11.1.12 Acabados del panel central

Finalmente, para el panel central, he escrito una página en HTML y un CSS para el diseño, y buscando he encontrado cosas nuevas que me han servido como la etiqueta <MARQUEE> que permite añadir letreros de texto en movimiento, y para el CSS la opción border-radius : 15px 15px 15px 15px; que he utilizado para dar unos bordes redondos a los <divs>, con los que he maquetado la web, y finalmente con <iframe> añadí RpiMonitor a mi panel central.

Tambien hice un logo mejor con Gimp gracias al trabajo con capas, y el resultado fue el siguiente:



RPi-Monitor

Version

Processor: **ARMv6-compatible processor rev 7 (v6l)**
Distribution: **Raspbian GNU/Linux 7 (wheezy)**
Kernel version: **Linux 3.12.20+ armv6l**
Firmware: **#685**
Package(s) to be: **0 upgraded, 0 newly installed**

Uptime

Raspberry Pi time: **03:17:28**
Uptime: **09** minutes **52** seconds

CPU

Loads: **0.28** [1min] - **0.30** [5min] - **0.24** [15min]
CPU frequency: **950MHz** Voltage: **1.35V**
Scaling governor: **ondemand**

Temperature

CPU Temperature: **51.382°C**

Memory

Used: **145.86MB** (**39.00%**) Available: **228.13MB** Total: **374.00MB**

Swap

Used: **0B** (**0.00%**) Free: **256.00MB** Total: **256.00MB**

SD card

/boot Used: **18.82MB** (**33.64%**) Free: **37.13MB** Total: **55.95MB**

Iframe de Rpi-Monitor

Antonio Mónaco Osado

11.2 Instalación de los clientes

Todas las aplicaciones cliente están disponibles para ser descargadas en el FTP de la Pi, que es accesible desde el manual de usuario en la MediaWiki.

11.2.1 Clientes GNU/Linux

Los clientes con un sistema operativo GNU/Linux ya traen preinstalados los clientes necesarios.

Cliente SAMBA: SAMBA.
Cliente XSANE: Simple Scan, (para diseñadores gráficos instalar xsane).
Cliente FreeRDP: Remmina
Cliente CUPS: CUPS
Cliente SSH: OpenSSH

11.2.2 Clientes Windows

Los clientes Windows disponen de algunas aplicaciones preinstaladas, otras como TwainSane debemos descargarlas.

Cliente SAMBA: SMB.
Cliente XSANE: TwainSANE.
Cliente RDP: Escritorio remoto.
Cliente CUPS: Las impresoras se instalan con doble click vía SAMBA.
Cliente SSH: putty, puttygen, winscp

11.2.3 Clientes MAC OS

Cliente SAMBA: SAMBA
Cliente XSANE: TwainSane
Cliente RDP: 2XRDP
Cliente CUPS: CUPS
Cliente SSH: OpenSSH

11.2.4 Clientes Android

Todas las aplicaciones Android se pueden descargar desde Google Play, solo debemos buscarlas e instalarlas, aunque disponemos de los enlaces a GooglePlay en el servidor FTP de la Raspberry, accesible desde el manual de la MediaWiki.

Las configuraciones son muy intuitivas, y los parámetros son los mismos que en Windows y Linux, aunque debemos configurar menos opciones.

Cliente SAMBA: AndSMB
Cliente XSANE: InSaneScanner
Cliente FreeRDP: 2XRDP
Cliente CUPS: Let's print droid
Cliente SSH: AndFTP, Connectbot

11.3 Configuración de los clientes

Los clientes deben configurarse correctamente, y algunas aplicaciones vienen ya instaladas en los Sistemas Operativos, y solo deberemos configurarlas.

Las que no estén por defecto, deberemos descargarlas del servidor FTP, instalarlas y finalmente configurarlas.

11.3.1 Clientes GNU/Linux

Cliente SAMBA: Podemos cambiar el dominio en smb.conf, para no escribirlo cada vez, y podemos acceder desde la opción conectar con el servidor del explorador de archivos Nautilus para guardar la conexión, e incluso añadir un marcador para tenerlo siempre accesible en un solo click.

Cliente XSANE: Simple Scan, (para diseñadores gráficos instalar xsane).

Simplemente debemos añadir la IP o nombre de host del PC en /etc/sane.d/net.conf y revisar que en /etc/sane.d/dll.conf el contenido sea ***net***.

Cliente FreeRDP: Debemos crear una sesión nueva en Remminna, simplemente introduciremos usuario, password y protocolo, en este caso RDP.

Con Remminna, podemos tunelizar la conexión RDP bajo SSH, creando así una conexión RDPS.

Cliente CUPS: Debemos ir al menú de configuración y a impresoras, y hacer click en conectar.

Finalmente añadimos la IP del servidor, y ya disponemos de las impresoras compartidas por el mismo.

Cliente SSH: Viene preinstalado por defecto, debemos crear un par de claves con ssh-keygen y copiar la publica con ssh-copy-id al authorized_hosts del servidor.

11.3.2 Clientes Windows

Cliente SAMBA: Debemos pulsar WindowsKey+R e introducir \\raspberrypi, y finalmente introducir nombre de usuario y password cuando lo pida al abrir un directorio.

Cliente XSANE: Después de instalar TwainSane, debemos darle la IP del servidor de escaneo en las preferencias del programa.

Cliente RDP: Con cliente de escritorio remoto, solo debemos escribir la IP de raspberry o su hostname y hacer click en conectar.

Cliente CUPS: Debemos acceder vía samba a Raspberry e instalar haciendo doble click la impresora que deseemos, seleccionando marca, modelo, tipo y fabricante.

Cliente SSH: Hay que configurar putty y WinSCP con los parámetros para acceder a la raspberry.

Con puttygen creamos un par de claves, y copiamos y pegamos el texto que nos muestra en pantalla al final del authorized_hosts del servidor, ya que exportando el archivo .pub no me ha servido el formato.

11.3.3 Clientes MAC OS

Solo he podido probar SAMBA y CUPS y el funcionamiento es idéntico que en Linux, aunque CUPS crea una cola local secundaria en local para la impresora remota.

11.3.4 Clientes Android

La configuración para los clientes Android es realmente sencilla e intuitiva, solo deberemos configurar los parámetros del servidor, es decir, la IP del servidor, y el usuario y password.

Solo cambia en AndSCP, ya que permite importar de la tarjeta SD una clave pública si lo deseamos.

Todos los servicios restantes se ofrecen vía navegador Web.

El servicio de cámaras de vigilancia de PiCam basado en Motion, solo funciona con el navegador Mozilla Firefox.



12 Hardware necesario

1 – Raspberry PI
2 – Alimentación, entrada 5V 1,5A
3 – Clavija RJ45.
4 – Controladora para puerto serial USB para administración de dispositivos.
5 – Hub USB alimentado para conectar las impresoras y scanners.
6 – Impresora multifuncion conectada por USB (para compartir el scanner y la impresora).

Antonio Mónaco Osado

13 Topología lógica

Red local
192.168.1.0/24

Rango DHCP 100-254
192.168.1.1

Rango DHCP 10-100
192.168.1.2

192.168.1.150

192.168.1.151

192.168.1.100

192.168.1.101

Rango DHCP 100-254: Es el rango DHCP del router.

Rango DHCP 10-100: Es el rango DHCP de la raspberry.

Antonio Mónaco Osado

14 Ejemplos de implementación del sistema (topología física)

14.1 Ejemplo 1: Todos los servicios en una Raspberry PI

Linksys WR54G

rpsolutions.com.es

raspinet.noip.me

Internet

DHCP Server

MySQL Server

Raspberry

Apache2 Server
S AMBA Server
DHCP Server
DNS Server
SSH Server
RDP Server
CUPS Server
XS ANE Server
Proxy cache SQUID
FTP Server
IPTABLES
PiCam

Impesora
Multifunción

Host 1

Host 2

Host 3

Host 4

Host 5

Host 6

Impesora WIFI

Antonio Mónaco Osado

14.2 Ejemplo 2: Mayor estabilidad repartiendo los servicios entre dos Raspberry PI

Linksys WR54G

rpsolutions.com.es

raspinet.noip.me

Internet

DHCP Server

MySQL Server

Raspberry

Apache2 Server
S AMBA Server
DHCP Server
DNS Server
SSH Server
RDP Server
Proxy cache SQUID
FTP Server
IPTABLES
PiCam

Impesora
Multifunción

Host 1

Host 2

Host 3

Host 4

Host 5

Raspberry

PiCam2
IPTABLES
XS ANE Server
CUPS Server
SSH Server
RDP Server

Host 6

Impesora WIFI

Antonio Mónaco Osado

15 Clientes compatibles

1 – Router Linksys WR54G
2 – Raspberry PI
3 – Cliente GNU/Linux
4 – Cliente Microsoft Windows
5 – Cliente Android
6 – Cliente MAC
7 – Cliente Iphone

Antonio Mónaco Osado

16 Pruebas finales en la Raspberry

Antes de dar por bueno el proyecto, es necesario realizar unas pruebas finales, así que lo he encendido durante 14 horas, con todos sus servicios funcionando y la PiCam grabando, teniendo en cuenta que Cron va ejecutando tareas a horas determinadas.

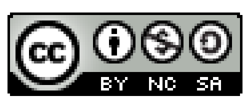
Tras la prueba los valores de la Pi eran los siguientes sin ventilador, con ventilador he logrado bajar ese valor a 39º-41º.

RPi-Monitor Status Statistics Configuration About

Version	Processor: ARMv6-compatible processor rev 7 (v6l) Distribution: Raspbian GNU/Linux 7 (wheezy) Kernel version: Linux 3.12.18+ armv6l Firmware: #679 Package(s) to be: 0 upgraded, 0 newly installed
Uptime	Raspberry Pi time: 13:18:51 Uptime: 14 hours 06 minutes 51 seconds
CPU	Loads: 8.35 [1min] - 10.82 [5min] - 9.80 [15min] CPU frequency: 700MHz Voltage: 1.20V Scaling governor: ondemand
Temperature	CPU Temperature: 62.679 °C
Memory	Used: 279.30MB (74.68%) Available: 94.70MB Total: 374.00MB
Swap	Used: 165.57MB (64.68%) Free: 90.43MB Total: 256.00MB
SD card	/boot Used: 18.80MB (33.61%) Free: 37.14MB Total: 55.95MB / Used: 5.44GB (37.34%) Free: 9.12GB Total: 14.56GB
Network	Ethernet Sent: 296.66MB ↑ Received: 43.43MB ↓

Monitorizando la Pi con Rpi-Monitor.

Antonio Mónaco Osado



18 Conclusión

Finalmente, he podido implementar todos los servicios que me propuse, y todos funcionan correctamente en clientes GNU/Linux, Windows, y Android. En Mac solo he podido probar SAMBA y CUPS, y en Iphone aún ninguno.

No ha sido fácil, pero al final he implementado mi proyecto de la mejor forma posible, sorteando todas las dificultades con las que he ido tropezando en el camino, y he demostrado (al menos a mí mismo) que el Hardware abierto y el Software libre y/o abierto son funcionales para la empresa, implementando muchas de las cosas aprendidas en clase estos dos años, aunque este proyecto se ha centrado mas en las asignaturas de Sistemas Operativos Monolugar, Sistemas Operativos en Red, Servicios de Red, Seguridad Informática y Aplicaciones Web.

A la hora de pensar en como debía diseñar este sistema, también aplique muchos conocimientos adquiridos en estos meses de prácticas de empresa.

Se debe tener al menos un nivel medio en Inglés para desarrollar este proyecto, ya que muchos de los manuales y tutoriales están disponibles únicamente en ese idioma.

Finalmente, pienso que este proyecto es totalmente funcional, aunque para mejorarlo, debería disponerse de una Raspberry con las aplicaciones web y los paneles centrales, y dejar una o mas Raspberry únicamente para actuar como cámara de videovigilancia, o repartir los servicios entre dos o mas Raspberry, siendo cámaras todas a la vez, como en el diagrama 2 del punto 14, para obtener fluidez y estabilidad, aunque por otro lado, quizás cuando salga al mercado la Raspberry modelo C con 1 GB de RAM esto ya no será necesario, y llegará en breves.

A lo largo de lo que queda de año.

Antonio Mónaco Osado – 2º SMX   Barcelona, 28 de Mayo de 2014